Abstract—Earlier static ID and Password are used which is vulnerable against eavesdropping and replay attack. To overcome this problem One Time Password technique is used which give different password each time. Earlier OTP is HOTP which is based on one way hash function SHA-1 and Ping Pong-128 stream cipher. In this paper we propose a method of generating OTP by using Genetic Algorithm with Elliptic Curve Cryptography.

Keywords—Ant Colony Algorithm, Genetic Algorithm, Elliptic Curve, Blowfish Algorithm and EABC

I. Introduction

As the name suggests that Network consist of large number of devices connected with each other for transferring the data and network security deals with the security about network means securely transferring the data on network. Network security deals with large number of devices to provide them secure access. User has to use its valid identification scheme to access the services provided by network. Unauthorized or unwanted user has no right to access the secure services provided on network. It provides security for both public which can be access by all users and private networks which can only be access by legitimate users. Network security performs authentication procedure by username and passwords. Network security is wide area of research today. Number of areas comes under their research is Mobile networks , Wireless Networks, Sensors Networks, Wireless sensors networks, Mobile Ad hoc networking, cryptography , Information security , Mobile security , wireless communication etc. Lot of research can be done on these areas, many algorithms are developed or modify , many security techniques are developed for providing the security in these areas. Cryptography is main tool in network security. It works on three attributes are Confidentiality, integrity and availability. Confidentiality means the authorized person is same as it claims to be , Integrity deals with the concept that data is received same as it is send by the sender and Availability deals with that the devices should available for providing network security. Cryptography use many Encryption and Decryption algorithm for dealing with confidentiality and Hash algorithms , MAC algorithms and Digital signature algorithms to deal with data integrity.

A. One Time Password

One time password generator is an algorithm that generate new random password every time. It works as a machine or algorithm that takes input from users and produce new password that is different from previously generated password. Network security deals with authenticate the user with id and password but this method is vulnerable to many attacks so for secure authentication every time new password is used whether the previous password is stolen or misplace. One time password generator is main element of One Time Password system used for generating the generating random passwords other elements of this system is client authentication and Server authentication. Popular OTP used are HOTP based on SHA-1,Hash algorithms used are MD4, MD5 but these are vulnerable to attacks. Another OTP is based on Ping Pong-128 stream cipher in which Ping Pong-128 algorithm is used to generate the random numbers.

One time password is secured because:
1. It can’t used twice or
2. It is not reversible to reach at source back.
It mainly deals with the two elements
1. Key
2. Counter

OTP system generates one password at a time and provides it to client for authentication.OTP send password to client by SMS service, by phone or by written. Password is secure by the application on client mobile.

In our work we purpose a One Time Password generator using Ant Colony Algorithm, Genetic algorithm with Elliptic Curve. Section 3 problem formulation. Chapter 4 explain about our work plan along with its timelines

II. Literature Review

A. Ant Colony Algorithm

It is the research field of Marco Dorigo et al. during its master thesis in 1991.After their research many version of ant colony algorithm are developed. It is the algorithm largely applied to many combinatorial optimization problems. After the ant colony developed as metaheuristic many variant of algorithm are developed.ACO depends upon the foraging behavior of ants of finding the food source. Ant finds the food source by traversing the shortest path and deposits the
chemical substance pheromone on it. Second time when another ant wants the food then it chooses that path having high pheromone value. Ants communicate with each other through chemical substance called pheromone [4].

1) Build the Solution: In this every ant builds their solution by traversing the graph. Ant on one node selects the node according to the probability:

\[ p_{xy}^k = \frac{(\tau_{xy})^\alpha(n_{xy})^\beta}{\sum y \in \text{allowed} (\tau_{xy})^\alpha(n_{xy})^\beta} \]

Where \( \tau_{xy} \) is the amount of pheromone on edge x, y. \( n_{xy} \) is the heuristic value which is describe as \( 1/d_{xy} \) (\( d_{xy} \) is the distance between edge x and y). \( \alpha \) and \( \beta \) are the relative influence on parameters \( \tau \) and \( n \). An ant repeat the previous step to find the valid tour of its each iteration and find the solution which is iteration best tour. All ant find their best tour after their each iteration, solution of each ant is compared and uses the best so far solution.

2) Update the Pheromone: After ant finds their tour they deposit pheromone substance on that path which is the process of updating the pheromone value. The pheromone values are more on paths which are traverse by more number of ants. For more effectiveness of more pheromone on good paths some amount of pheromone is removed from the path to avoid the bad quality solution. So pheromone updating consists of pheromone evaporation as negative feedback and pheromone updating as positive feedback. The rule is:

\[ \tau_{xy} \leftarrow (1 - \rho)\tau_{xy} + \sum_k \Delta\tau_{xy}^k \]

Where \( (x,y) \in L \) is set of all edges , \( \rho \) is the evaporation rate , \( k \) is number of ants. Pheromone update by ant \( k \) is

\[ \Delta\tau_{xy}^k = \begin{cases} Q/L_k & \text{if ant } k \text{ uses curve } xy \text{ in its tour} \\ 0 & \text{otherwise} \end{cases} \]

Where \( L_k \) is the tour length constructed by ant \( k \) and \( Q \) is a constant.

B. Genetic Algorithm

Genetic Algorithm is the research field of John Holland based on the principle of natural genetics. It maintains some initial population and generate a solution from that population by applying their operators of selection, crossover and mutation to find a optimize solution GA use the variables in the form of binary string \{0, 1\} because it is optimal solution. In GA the population is choose called as chromosomes and from that population offspring is produced by applying the crossover and mutation operator. Parents choose from the population for producing offspring are according to their fitness value and fitness value of offspring is also calculated so that less fit offspring should be deleted [9][10].

Stages in Genetic Algorithm

The different stages of GA [9] are:

1) Initial Population of GA: First step in GA is to select the population of chromosomes. The fitness function of all these chromosomes is calculated. According to the different research the initial population size should be random. Other possible solution is the ‘seeding’ of initial population i.e. according to some research initial population of GA should be the output of some other heuristic technique produces good results.

2) Selection of Chromosomes: The parents or chromosomes that are to be select for reproduction is choose according to their fitness value. Most common method is choosing according to the fitness so that more fit parent has more chance for reproduction.

3) Crossover: After selecting the parents of high fitness value next step is to perform the crossover operation. In this few chromosomes of one parent is replace with other parent so that new offspring should be generated having some characteristics similar to that of their parents. Example we have two parents P1 and P2 as:

\[
P1 = 10010001 \\
P2 = 11011000
\]

And after crossover the offspring’s are as: X3 = 10011000 and X4 = 11010001.

4) Mutation: The next step after crossover is mutation. In some cases there is no need for crossover directly mutation is performed. In case after crossover two same offspring’s are produced then random bits from one offspring is mutated to produce different offspring. Suppose we have offspring X3 = 1011000 and we want to mutate its 4 and 8 bit then after mutation new offspring is X5 = 10001001.

C. Elliptic Curve Algorithm

An elliptic curve E takes the general form as:

\[ E: y^2 = (x^3 + ax + b) \mod P \]

Where a, b are in the appropriate set (rational numbers, real numbers, integers mod p, etc.) and x, y are elements of the
finite field $\text{GF}(p)$, satisfying $4a^3 + 27b^2 \neq 0 \pmod{p}$.\textsuperscript{(5)} and $p$ is known as modular prime integer making the elliptic curve finite field\textsuperscript{(5)}.

There are two basic group operations on elliptic curve which are point addition and point doubling.

1) **Point Addition:** Addition means that given two points $E$ and their coordinates, $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ belongs to $E(\text{GF}(p))$, we have to compute the coordinates of a third point $R$ such that:

$P + Q = R$

$\left(x_1, y_1\right) + \left(x_2, y_2\right) = \left(x_3, y_3\right)$

This is the case where we compute $R = P + Q$ and $P \neq Q$. Point R’s coordinates $(x_3, y_3)$ also belongs to $E(\text{GF}(p))$.

- $\lambda = \frac{y_p - y_q}{x_p - x_q}$
- $x_r = \left[\lambda^2 - x_p - x_q\right] \pmod{p}$
- $y_r = \left[-y_p + \lambda(x_p - x_r)\right] \pmod{p}$

2) **Point Doubling:** Point doubling is the addition of a point $P$ on $E$ to itself to obtain another point $R$. This is the case where we compute $P + Q$ but $P = Q$. Hence we can write $R = P + P = 2P$.

- $\lambda = \frac{3x_p^2 + \alpha}{2y_p}$
- $x_r = \left[\lambda^2 - 2x_p\right] \pmod{p}$
- $y_r = \left[-y_p + \lambda(x_p - x_r)\right] \pmod{p}$

**D.BLOWFISH**

Bruce Schneider designed blowfish in 1993 as a fast, free alternative to existing encryption algorithm. Since then it has been analyzed considerably, and it is slowly gaining acceptance as a strong encryption algorithm. The Blowfish algorithm has many advantages. It is suitable and efficient for hardware implementation and no license is required\textsuperscript{(16)}.

The elementary operators of blowfish algorithm include table lookup, addition and XOR. The table includes four S-boxes and a P-array. Blowfish is a cipher based on feistel rounds, and the design of the F-function used amounts to a simplification of the principle used in DES to provide the same security with greater speed and efficiency in software.

Blowfish is a 64 bit block cipher and is suggested as a replacement for DES. Some specifications of Blowfish algorithm are as follows:

1. A 64 bit blocks cipher with a variable key length.
2. There is a P-array and four 32-bit S-boxes. The P-array contains 18 of 32-bit sub keys, while each S-box contains 256 entries.
3. The algorithm consists of two parts: a key-expansion part and a data-encryption part.
4. Key expansion part converts a key of at most 448 bits into several sub keys array totaling 4168 bytes.
5. The data encryption occurs via a 16-round Feistel network. Each round consists of a key dependent permutation and a key and a data dependent substitution.
6. All operators are XORs and additions on 32-bit words.
7. The input is 64 bit data element.

The process of Sub key generation is illustrated as follows:

1. Initialize P array and S boxes with Hexadecimal digits of Pi.
2. XOR P-array with the key bits (i.e., P1 XOR (first 32 bits of key), P2 XOR (second 32 bits of key).
3. Use the above method to encrypt the all-zero string.
4. This new output is P1 and P2.
5. Encrypt the new P1 and P2 with the modified sub keys.
6. This new output is now P3 and P4.
7. Repeat the above steps until we get all the elements of P array i.e. P1, P2.

The encryption algorithm for Blowfish is illustrated as follows:

<table>
<thead>
<tr>
<th>Table I: Blowfish Encryption algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Divide X into two 32-bit Halves: XL, XR</td>
</tr>
<tr>
<td>2. For I = 1 to 16</td>
</tr>
<tr>
<td>XL = XL $\oplus$ Pi</td>
</tr>
<tr>
<td>XL = F(XL) $\oplus$ XR</td>
</tr>
<tr>
<td>Swap XL AND XR</td>
</tr>
<tr>
<td>3. Swap XL AND XR (undo the last step)</td>
</tr>
<tr>
<td>4. XR = XR $\oplus$ P17</td>
</tr>
<tr>
<td>5. XL = XL $\oplus$ P18</td>
</tr>
<tr>
<td>6. Concatenate XL and XR</td>
</tr>
</tbody>
</table>
III. Problem Formulation

After studying the papers and how the OTP works, we formulate our problem as One Time Password generator using Genetic Algorithm with Elliptic Curve Algorithm, Blowfish algorithm \(^{(16)}\) and EABC \(^{(17)}\).

**A. NEW APPROACH**

We explain our new approach as:

1. When user login servers take its Email ID and Password and authenticate the user.
2. Server simply encrypts the Id and password and gives that output to OTP generator.
3. OTP generator starts its work. OTP selects two alphabets from encrypted data and use genetic algorithm.
4. By using genetic operators we create random 8 alphabets from that two alphabets and suppose it is Random key.
5. We have to select 8 alphabets from encrypted output assume it as ID.
6. Now we have two keys of 8 alphabets. One is Random key and second is Identification ID.
7. Divide Identification ID and Random password into two halves of 4 alphabets. i.e.
   \[
   \text{pswd}_L = X_1X_2X_3X_4 \quad \text{pswd}_R = X_5X_6X_7X_8
   \]
   \[
   \text{ID}_L = Y_1Y_2Y_3Y_4 \quad \text{ID}_R = Y_5Y_6Y_7Y_8
   \]
8. Take random point from elliptic curve which satisfy equation of elliptic curve and convert it into binary form of 8 bits.
9. Now according to the binary value perform steps.
   - If \(b[i] == 0\) perform
     \[
     K_L = \text{pswd}_L \text{ (OR operation) } ID_L
     \]
     \[
     K_R = \text{pswd}_R \text{ (OR operation) ID}_R
     \]
   - Else if \(b[i] == 1\) perform
     \[
     K_L = \text{pswd}_L \text{ (OR operation) ID}_L
     \]
     \[
     K_R = \text{pswd}_R \text{ (OR operation) } F(ID_R)
     \]
   Where \(F(\text{ID}_R) = \text{Product between the ID}_R \text{ and Any random point in elliptic curve.}\)
10. Merge \(K_L\) AND \(K_R\) is equal to K.
11. Merge \(F(\text{ID}_R)\) and \(\text{ID}_L = \text{ID}\).
12. Find the product of ID with any private key.
13. We have now 8 Random passwords which we are store in database and newly generated identification No. (ID).
14. Next time when user login then that Identification No. (ID) is given to OTP generator for generating password.

IV. Results and Discussion

The OTP generator algorithm was successfully implemented in JAVA with an environment ubuntu (64-bit) Operating System, Pentium Processor and Memory of 2GB RAM, 320 GB Hard Disk. The Screenshots describe below shows the working of OTP algorithm.

A. Screenshots: The user provide the random passphrase of 10 digit at the time of OTP generation and this 10 digit random key is the unique ID value which is provided by authenticated sites to its users.

![OTP Screenshot](image1.jpg)

Fig. 1. OTP is generated by using saved ID value.

As shown in Fig. 1, user give the random key which is minimum of 10 alphabets for generating the new OTP. This 10 digit random alphabets may be the unique ID value provided by authenticated sites to its user. Before creating the password the user has to authenticate with this unique ID to server and then use this ID value for creating new OTP every
time. When user put this unique ID value then after clicking “show stepwise generation of next OTP” button the procedure of OTP generation is start. Firstly generator picks the username and password of user as shown in Fig.2. Next the generator picks the encrypted data of username and password from which it gives two alphabets to genetic algorithm for creating the 8 random alphabets and choose the most fit child of 8 bits as shown in Fig.3. After that generator pick the 8 random bits from the ECC curve and perform the same operation as explain in Point 10 of New approach under section in Fig.4. Every time new key and ID value is generated. At last the hash code is created from new ID and key value shown in Fig.5. New Id is save in User database and the Key is given to the user as a new OTP as in Fig.6.

B. Efficiency and Security analysis

1) Security: We are using two cryptography algorithms in our algorithm which is computationally efficient and provide more security. There are 15 elliptic curves available of varying sizes in which 10 are known as binary fields and 5 are prime fields. Elliptic curve also use for cryptography same as another encryption algorithms and it uses the key size of 80, 112, 128, 192 and 256 bits. Next Modify Blowfish algorithm provide the security by using F function according to the flag value and generate different cipher text for different plaintext. It consumes less time as compared to the Blowfish algorithm. In proposed algorithm the system choose random bits from the elliptic curve and based on the values of that bits either it is 1 or 0 bit perform the F function which produce the different keys every time the process starts. As in earlier OTP systems one password is generated by calculating the hash function of previous password but in this ID of user is used next time for generating password which is kept in secure place at server and every time hash code of new Id and key is created. Secondly we use the concept of failed login and deletion of fake ID depend upon the status of a person as user is blocked after three attempts if he enter wrong password every time and server delete the ID of user by checking its status if he is not login in their account few days after registration then its status become 0 so server delete its account.

---

Fig. 2. Encryption output and genetic algorithm is shown.

Fig. 3. Output of genetic and encryption algorithm is shown.
Fig. 4. Random Bits from ECC is selected and new key is generated.

Fig. 5. Hash of new Id and value is generated and new key and Id is shown below in screen.

2) Efficiency: Efficiency of our algorithm is measured on various parameters shown in Fig. 7.

Start and End Time: The time consume by our algorithm is less as compared to other algorithms. It consumes very less start and end time for their operation. The start time of our algorithm is 54091352030 (nanoseconds) and end time is 5409151677476 (nanoseconds) so the total time consume by the algorithm is 0000017149446 (nanoseconds) which shows that it consumes less time as compared to other algorithms.

Delay: As we see above the start and end time consume by our algorithm so the total delay in our algorithm is 17149446 (nanoseconds) which is less than the delays of existing algorithms. Which shows that it is more efficient.

Throughput: The throughput provided by our algorithm is large as compared to others. Our approach provides the throughput value of 5 which is more than existing Algorithms. Which shows that proposed algorithm provide more efficiency.

Fig. 6. Random key for user is created
One time password is an efficient technique that generate random password each time for users. If user lost their pervious password then there is no need of worry for them because OTP give them new password for each session. OTP prevent user id from replay or eavesdropping attack. Earlier OTP is generated using HMAC , One way hash function and Ping Pong stream cipher , in which input is given to OTP generator as challenge and it generate random password. In our work we propose a method of generating OTP generator using Genetic algorithm with elliptic curve algorithm. In future more how to provide more security in this approach and on secure authentication between client and server.

Acknowledgment

The author are thankful to Ms. Anu garg ,Assistant Professor at Lovely Professional University for giving good suggestions during our work that enable us to present this work and author also thankful to her Father for encouraging her.

References

[3] Bayalagmas Davaanaym, Young Sil Lee, HoonJaeLee, SangGon Lee and HyoTeak Lim, ” A Ping Pong One-Time-Password system in Java application”.
[7] Dewen Zeng, Qing He, Bin Leng, Weimin zheng, Hongwei Xu, Yiyu Wang, Guan,” An Improved Ant Colony Optimization Algorithm Based on Dynamically Adjusting Ant Number”.
[8] Zhiguo Liu , Tao Liu, Xiu Gao,” An Improved Ant Colony Optimization Algorithm Based on Pheromone Backtracking”.
[9] Colin Reeves, School of Mathematical and Information Sciences,” GENETIC ALGORITHMS”.
[17] Hamid Mehdi, “EABC: Data Encryption Method Based on Circle”.
Saini et al., International Journal of Advanced Research in Computer Science and Software Engineering 4(6), June - 2014, pp.655-662


[22] D. M’Raihi, M. Bellare UCSD, F. Hoornaert VascoD. Naccache Gemplus O.Ranen
Aladdin December 2005 Network Working Group, Request for Comments: 4226,Category: Informational,”
HOTP: An HMAC-Based One-Time Password Algorithm”.

Category: Informational ISSN: 2070-1721 Diversinet Corp.” TOTP: Time-Based One-Time Password Algorithm”.

Bellcore February 1998,”Request for comments : 2289 A One-Time Password System”. 